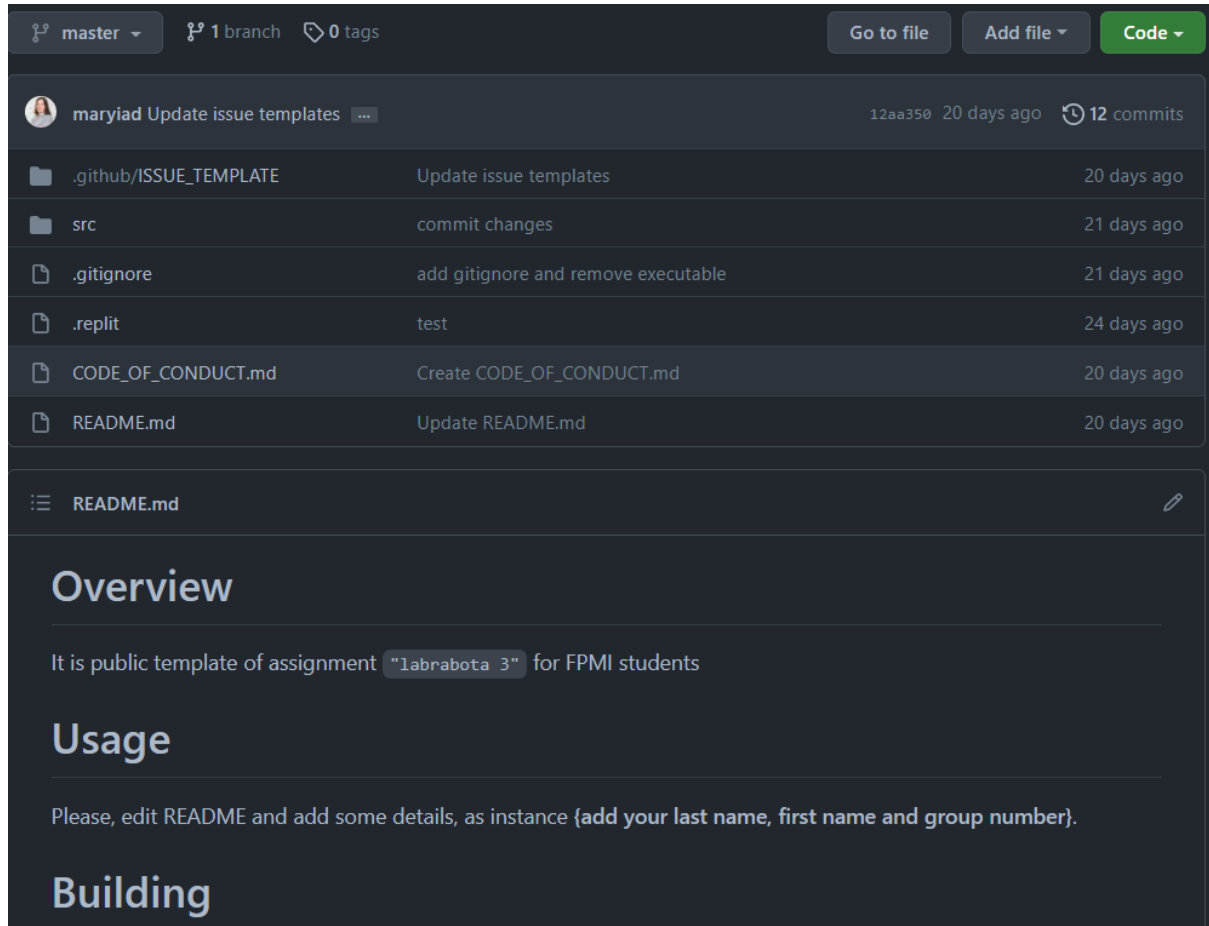
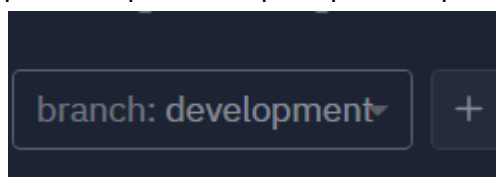


ЗАДАНИЕ 1. КОНСОЛЬНЫЕ ПРИЛОЖЕНИЯ В REPL.IT

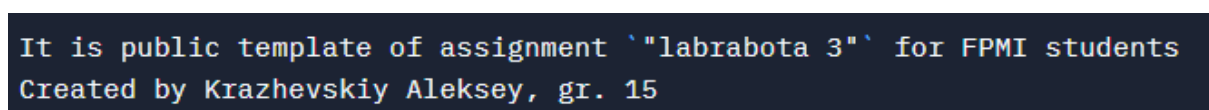
2. Создайте репозиторий из шаблона репозитория согласно рекомендациям руководства из п. 1 текущего задания. В качестве имени репозитория введите имя согласно шаблону lab3-task1-gr13-petrov, указав корректно номер своей группы и фамилию вместо petrov.



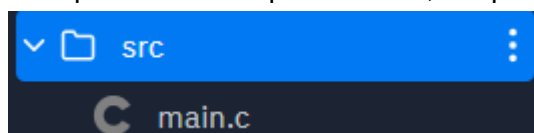
4. Создать ветку в репозитории для данной задачи согласно Требования к репозиториям, например development, и переключиться в неё.



5. Отредактируйте файл Readme в созданном проекте, изменив текст в разделах Overview, Usage, Building. Добавьте в Readme свою фамилию, имя и номер подгруппы вместо текста {add your last name, first name and group number}. Закоммитить изменения.



6. Переименовать файл hello.c, например в main.c.



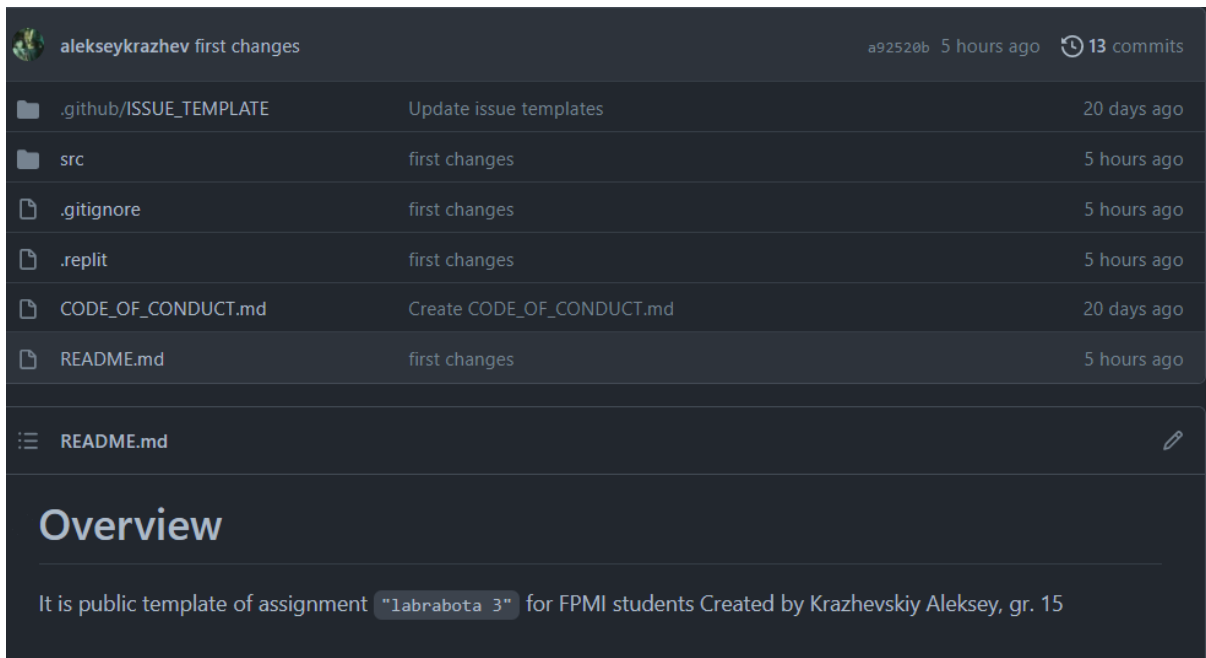
7. Отредактировать файл .replit и указать название файла, который компилируется, например main.c, и имя бинарного файла

```
1 language = "c"
2 run = "gcc -o b_file /home/runner/project2/src/main.c"
```

8. Отредактировать .gitignore и указать имя бинарного файла, который должен получиться в результате labrabota3-1 и который не должен быть опубликован в репозиторий, т. е. Проигнорирован при публикации изменений.

```
# Executables
*.exe
*.out
*.app
*.i*86
*.x86_64
*.hex
b_file
```

9. Закоммитить изменения и опубликовать их в репозиторий на github.



The screenshot shows a GitHub repository page for user 'alekseykrazhev' with the title 'first changes'. The repository has 13 commits and was updated 5 hours ago. A list of files and their commit history is shown:

File	Commit Message	Time
.github/ISSUE_TEMPLATE	Update issue templates	20 days ago
src	first changes	5 hours ago
.gitignore	first changes	5 hours ago
.replit	first changes	5 hours ago
CODE_OF_CONDUCT.md	Create CODE_OF_CONDUCT.md	20 days ago
README.md	first changes	5 hours ago

Below the file list, there is a section for 'README.md' with an 'Overview' heading. The overview text reads: 'It is public template of assignment "labrabota 3" for FPMI students Created by Krazhevskiy Aleksey, gr. 15'.

10. Сделано

11-12, 14. Изменения закоммичены, массив заволяется рандомно

13. Провести тесты (контрольный расчет) для проверки работоспособности алгоритма решения задачи и включить их в отчёт.

```

133 136      27 115      143 85      136 42      99 121      62 127      140 109
  113 76      90 126      22 136      11 68      117 129      32 80      62 73
  67 85      29 102      72 108      119 117      43 106      61 142      129 123
    21 119      134 37      98 74      65 120
113 76      91 80      56 123      62 20      46 31      105 75      134 27      86
  5      46 129      13 107      24 45      132 45      14 117      134 14      93 5
0      37 108      126 128      38 84      3 101      104 49      132 60      26 18
    139 112      76 36      144 89
First

```

```

43 136      37 115      113 85      66 42      29 121      2 127      90 109      8
3 76      80 126      72 136      31 68      47 129      2 80      82 73      127 8
5      89 102      122 108      129 117      113 106      51 142      29 123      6
1 119      104 37      118 74      135 120
13 76      91 80      56 123      62 20      96 31      5 75      84 27      36 5
    46 129      13 107      24 45      82 45      14 117      34 14      43 50
  87 108      76 128      88 84      3 101      54 49      32 60      76 18      39
112      26 36      94 89
First

```

```

83 136      77 115      93 85      86 42      49 121      62 127      90 109      6
3 76      40 126      72 136      11 68      67 129      82 80      62 73      67 8
5      29 102      22 108      69 117      93 106      11 142      29 123      21 1
19      84 37      98 74      15 120
13 26      91 480      56 373      62 170      96 281      5 425      84 327      3
6 5      46 229      13 357      24 395      82 45      14 367      34 364      43
250      87 308      76 178      88 84      3 151      54 399      32 60      76 36
8      39 12      26 86      94 39
second

```

15. Переключиться в репозиторий на github и, используя pull request, слить изменения из ветки development в ветку master

master
 2 branches
 0 tags
 Go to file
Add file
Code

alekseykrazhev Merge pull request #1 from alekseykrazhev/development
 dd48b7b 13 seconds ago
 16 commits

📁 .github/ISSUE_TEMPLATE	Update issue templates	20 days ago
📁 src	Completed task	1 minute ago
📄 .gitignore	first changes	5 hours ago
📄 .replit	first changes	5 hours ago
📄 CODE_OF_CONDUCT.md	Create CODE_OF_CONDUCT.md	20 days ago
📄 README.md	first changes	5 hours ago
📄 main	Completed task	1 minute ago

ЗАДАНИЕ 2. ИЗУЧАЕМ КУРС GitHub Actions: Hello World.

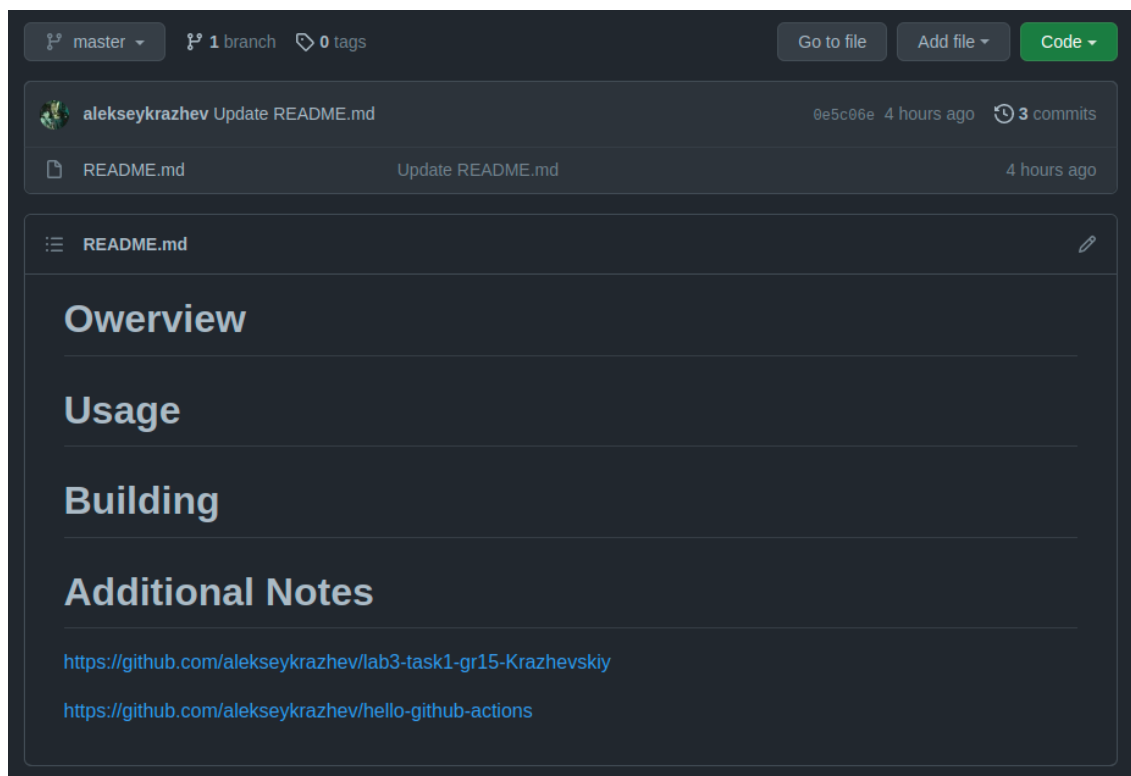
Progress

7 of 7

Course steps		...
✓	Add a Dockerfile	Completed 3 minutes ago
✓	Add an entrypoint script	Completed 3 minutes ago
✓	Add an action.yml file	Completed 2 minutes ago
✓	Start your workflow file	Completed 2 minutes ago
✓	Run an action from your workflow file	Completed a minute ago
✓	Trigger the workflow	Completed a few seconds ago
✓	Incorporate the workflow	Completed a few seconds ago

ЗАДАНИЕ 3. ИСПОЛЬЗУЕМ СТРУКТУРЫ ДЛЯ ПРЕДСТАВЛЕНИЯ ДАННЫХ

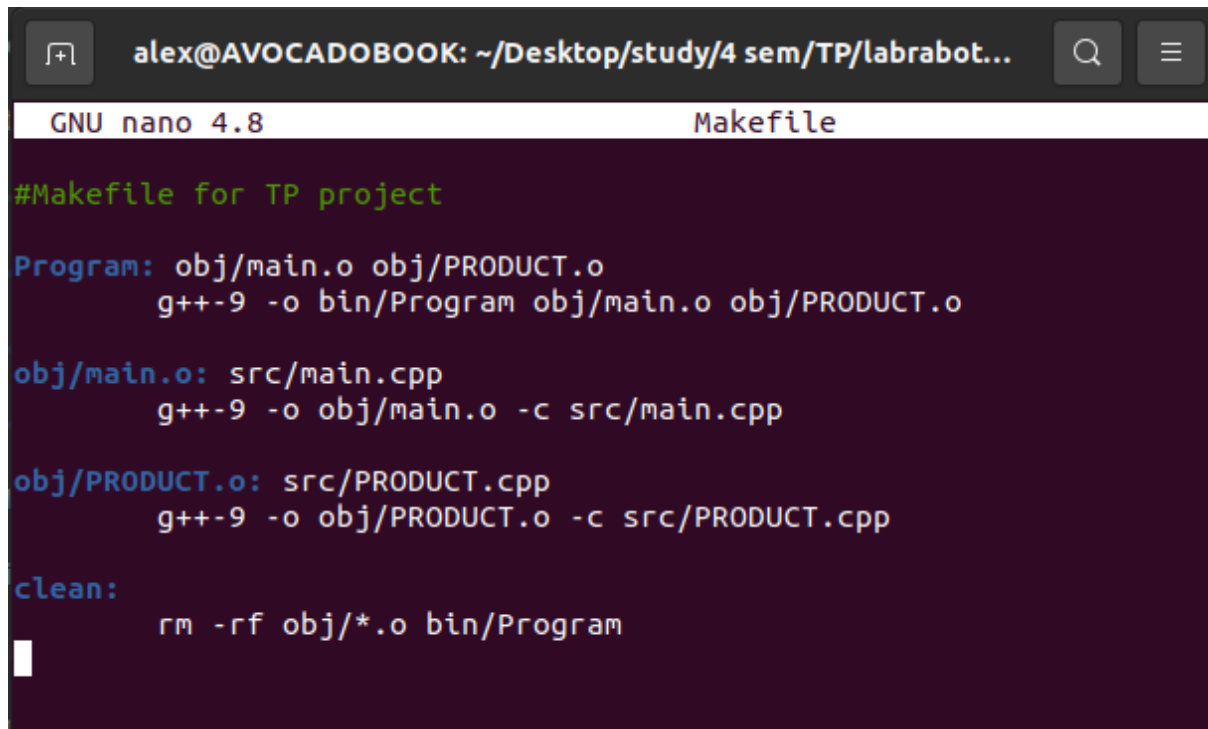
2.



Makefile выполнение:

```
alex@AVOCADOBOOK: ~/Desktop/study/4 sem/TP/labrabota3-z3-gr15-alekseykrazhev$ make Program
g++-9 -o obj/main.o -c src/main.cpp
gcc -o obj/PRODUCT.o -c src/PRODUCT.cpp
g++-9 -o bin/Program obj/main.o obj/PRODUCT.o
```

Makefile:



```
alex@AVOCADOBOOK: ~/Desktop/study/4 sem/TP/labrabot...
GNU nano 4.8 Makefile

#Makefile for TP project

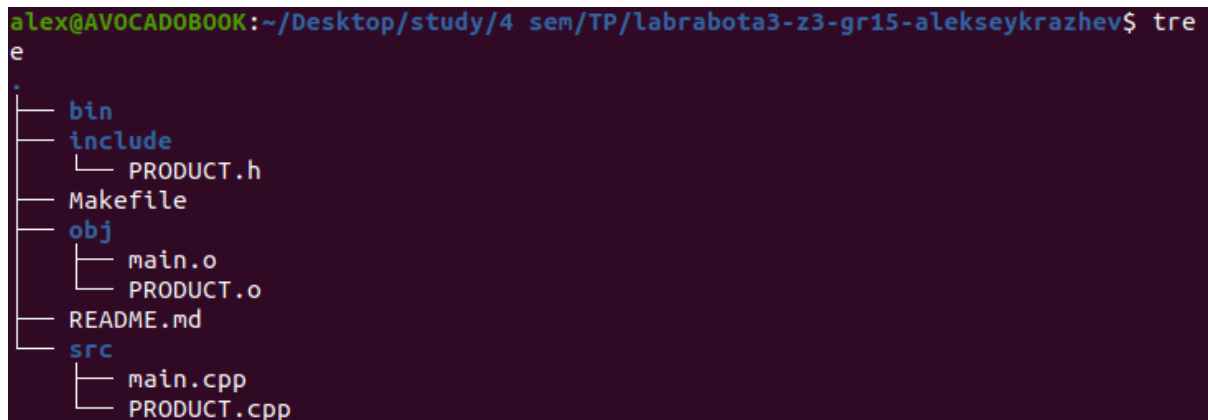
Program: obj/main.o obj/PRODUCT.o
        g++-9 -o bin/Program obj/main.o obj/PRODUCT.o

obj/main.o: src/main.cpp
        g++-9 -o obj/main.o -c src/main.cpp

obj/PRODUCT.o: src/PRODUCT.cpp
        g++-9 -o obj/PRODUCT.o -c src/PRODUCT.cpp

clean:
        rm -rf obj/*.o bin/Program
```

Директория:



```
alex@AVOCADOBOOK: ~/Desktop/study/4 sem/TP/labrabota3-z3-gr15-alekseykrazhev$ tree
.
├── bin
├── include
│   └── PRODUCT.h
├── Makefile
├── obj
│   ├── main.o
│   └── PRODUCT.o
├── README.md
└── src
    ├── main.cpp
    └── PRODUCT.cpp
```

main.cpp:

```
alex@AVOCADOBOOK: ~/Desktop/study/4 sem/TP/labrabot...
GNU nano 4.8 src/main.cpp
#include <iostream>
#include "/home/alex/Desktop/study/4 sem/TP/labrabota3-z3-gr15-alekseykrazhev/i

int main(){
    std::vector<PRODUCT> vec = init(10);
    two_days(vec);
    same_sup(vec);
    max_price(vec);
    date_up(vec);
    return 0;
}
```

PRODUCT.cpp:

```
alex@AVOCADOBOOK: ~/Desktop/study/4 sem/TP/labrabot...
GNU nano 4.8 src/PRODUCT.cpp
    PRODUCT pr;
    std::cout << "Enter name:\n";
    std::cin >> pr.name;
    std::cout << "Enter price:\n";
    std::cin >> pr.price;
    std::cout << "Enter date:\n";
    std::getline(std::cin, pr.date);
    std::cout << "Enter till its good:\n";
    std::cin >> pr.good;
    std::cout << "Enter amount:\n";
    std::cin >> pr.amount;
    std::cout << "Enter supplier:\n";
    std::cin >> pr.supplier;
    vec.push_back(pr);
}
return vec;
}

void two_days(std::vector<PRODUCT> vec){
    for (auto i : vec) {
```

Тест программы:

```
alex@AVOCADOBOOK: ~/Desktop/study/4 sem/TP/labrabot...
Products with two days left:
hill 458 23 05 2013 2 5 belgosstrah
car 5600 02 12 1995 2 1 derevnya

Count for suppliers:
alal count = 2
sup count = 4
derevnya count = 2
belgosstrah count = 2

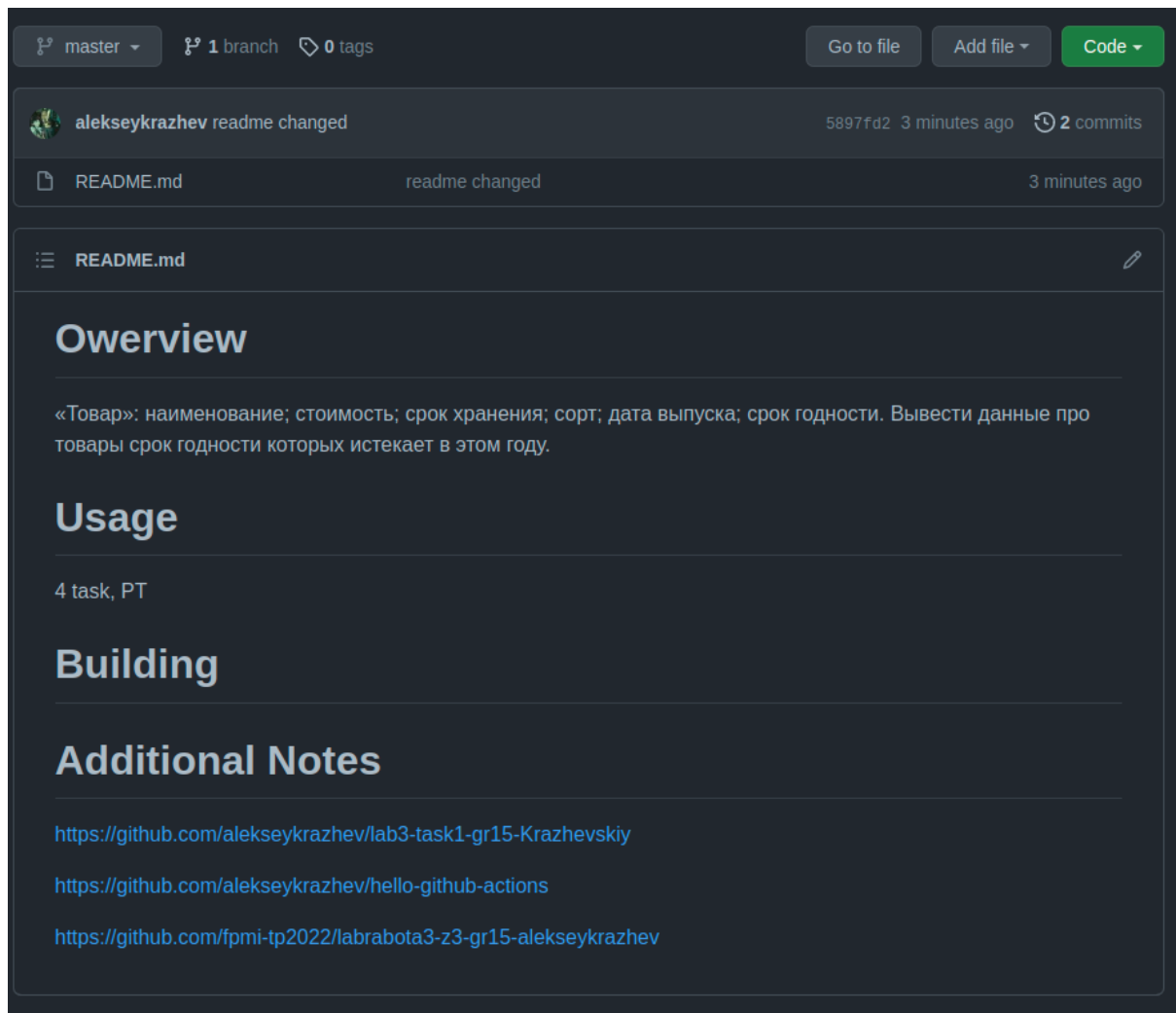
Max price:
meat 44 23 03 2022 6 4 sup

Sorted:
car 5600 02 12 1995 2 1 derevnya
lemon 569 18 02 1999 -100 3 belgosstrah
bread 45 02 02 2002 4568 55 alal
tree 896 03 12 2002 5 56 sup
water 578 14 11 2008 45 456 sup
roses 4567 02 12 2012 5 5 alal
hill 458 23 05 2013 2 5 belgosstrah
milk 45 25 02 2022 5 5 derevnya
meat 44 23 03 2022 6 4 sup
fish 895645 03 12 20021 -1 45 sup
alex@AVOCADOBOOK:~/Desktop/study/4 sem/TP/labrabota3-z3-gr15-alekseykrazhev$
```

master	2 branches	0 tags	Go to file	Add file	Code
alekseykrazhev Merge pull request #1 from fpmi-tp2022/development 13ddbfb now 5 commits					
bin	task completed	1 minute ago			
include	task completed	1 minute ago			
obj	task completed	1 minute ago			
src	task completed	1 minute ago			
Makefile	task completed	1 minute ago			
Makefile.save	task completed	1 minute ago			
README.md	Update README.md	7 hours ago			

ЗАДАНИЕ 4. ЗАПИСЬ И ЧТЕНИЕ ТЕКСТОВЫХ ФАЙЛОВ

1-2:



4. При работе над заданием следуйте правилам создания веток, изложенным в документации Требования к репозиториям.

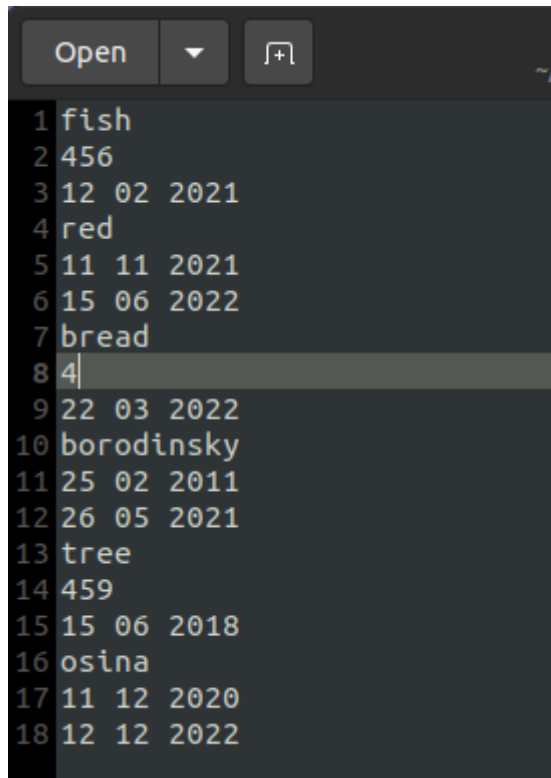
```
alex@AVOCADOBOOK:~/Desktop/study/4 sem/TP/labrabota3-z4-gr15-alekseykrazhev$ git  
branch  
* development  
master
```

5. Создайте структуру проекта согласно Структура проекта согласно модели КИС и правил сборки.

6. Используя редактор nano, напишите программу для обработки текстовых файлов. Файл вывода при тесте:

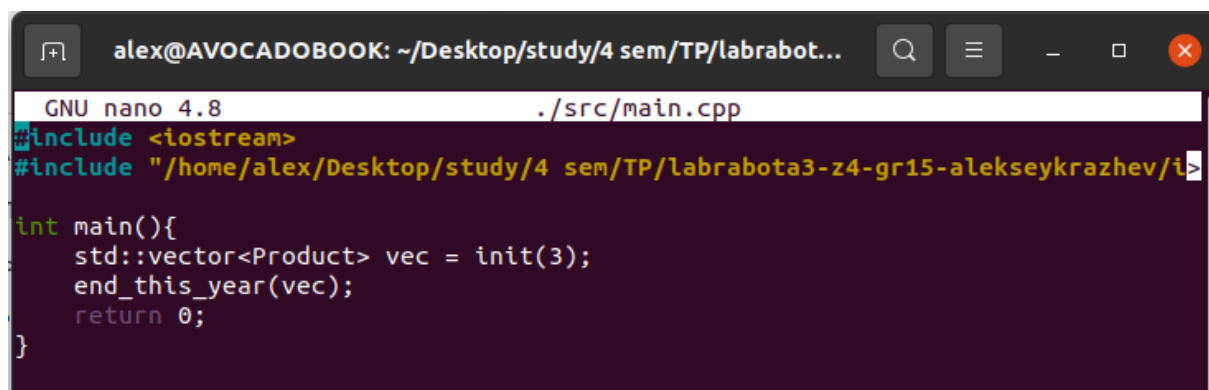
The screenshot shows a nano text editor window. The title bar indicates the file is 'out.txt' and its location is '~/.Desktop/study/4 sem/TP/labrabota3-z4-gr15-alekseykrazhev/doc'. The editor contains the following text:
1 Products ending in 2022:
2
3 fish 456 12 02 2021 red 11 11 2021 15 06 2022
4
5 tree 459 15 06 2018 osina 11 12 2020 12 12 2022

Файл ввода для теста:



```
1 fish
2 456
3 12 02 2021
4 red
5 11 11 2021
6 15 06 2022
7 bread
8 4
9 22 03 2022
10 borodinsky
11 25 02 2011
12 26 05 2021
13 tree
14 459
15 15 06 2018
16 osina
17 11 12 2020
18 12 12 2022
```

main.cpp:



```
alex@AVOCADOBOOK: ~/Desktop/study/4 sem/TP/labrabot...
GNU nano 4.8 ./src/main.cpp
#include <iostream>
#include "/home/alex/Desktop/study/4 sem/TP/labrabota3-z4-gr15-alekseykrazhev/i

int main(){
    std::vector<Product> vec = init(3);
    end_this_year(vec);
    return 0;
}
```

Product.h:

```
alex@AVOCADOBOOK: ~/Desktop/study/4 sem/TP/labrabot...
GNU nano 4.8 ./include/Product.h
#include <vector>
#include <string>
#include <sstream>
#include <fstream>
#include <iostream>

struct Product{
    std::string name;
    std::string price;
    std::string store;
    std::string sort;
    std::string date;
    std::string good;
};

std::vector<Product> init(int value);

void end_this_year(std::vector<Product>);
```

Структура папок:

```
alex@AVOCADOBOOK:~/Desktop
├── bin
│   └── Program
├── doc
│   ├── in.txt
│   └── out.txt
├── include
│   └── Product.h
├── Makefile
├── obj
│   ├── main.o
│   └── Product.o
├── README.md
└── src
    ├── main.cpp
    └── Product.cpp
```

Makefile:

```
alex@AVOCADOBOOK: ~/Desktop/study/4 sem/TP/labrabot...
GNU nano 4.8 Makefile

#Makefile task 4

Program: obj/main.o obj/Product.o obj
        g++-9 -o bin/Program obj/main.o obj/Product.o

obj/main.o: src/main.cpp
        g++-9 -o obj/main.o -c src/main.cpp

obj/Product.o:
        g++-9 -o obj/Product.o -c src/Product.cpp

clean:
        rm -rf obj/*.o bin/Program
```

Сборка:

```
alex@AVOCADOBOOK:~/Desktop/study/4 sem/TP/labrabota3-z4-gr15-alekseykrazhev$ make
make Program
g++-9 -o obj/main.o -c src/main.cpp
g++-9 -o obj/Product.o -c src/Product.cpp
g++-9 -o bin/Program obj/main.o obj/Product.o
alex@AVOCADOBOOK:~/Desktop/study/4 sem/TP/labrabota3-z4-gr15-alekseykrazhev$
```

3-5 выполнено

Контрольные вопросы

1. Примерами облачных IDE являются Repl.it., Cloud 9, ShiftEdit, Codeanywhere и т.д.
2. Подключить репозиторий в repl.it. можно с помощью create, после чего выбирается создание по шаблону или импортировать через гит (import from github), выбрав нужный репозиторий.
3. Сборкой из нескольких исходных файлов можно создать работающую программу с помощью их компиляции и линковки.
4. Утилиты для сборки: make, gcc, CMake.
5. Makefile - набор инструкций для утилиты make, с помощью которой можно собрать программу из исходных файлов. То есть Makefile содержит все необходимые сценарии для make.

6. Целью в Makefile является тот файл, который должен получиться в итоге. Как пример, в третьем задании целью Makefile был файл Program.
7. Связкой в Makefile распределяют цели, зависимости и правила. Как пример, в третьем задании возьмем связку: `obj/main.o: src/main.c`.
8. Зависимости исходных файлов. Как пример, рассмотрим зависимость из четвертого задания: `obj/main.o obj/Worker.o`.
9. Правилom является команда, которая нужна для того, чтобы выполнять цель, т.е. сгенерировать конечный. Из чего, как и что. Пример из четвертого задания: `gcc -o bin/Program obj/main.o obj/Worker.o`
10. Макроопределение задает некоторое имя (метку), которая будет использоваться вместо нескольких других в сценарии (для краткости, чаще всего). Возьмем пример из четвертого задания: `mac = gcc`. тогда `$(mac) -o bin/Program obj/main.o obj/Worker.o`
11. Для очистки проекта используется связка `clean: rm -rf <>`
12. КИС удобное распределение всех исходных файлов по папкам, а также, например, отделение заголовков от главного.
13. Общепринятые правила по оформлению программного кода. Рекомендуется следовать стандартам кодирования, чтобы облегчить читаемость кода для других программистов, которые будут как-либо взаимодействовать с Вашим кодом.
14. Проект должен состоять из нескольких папок, таких как `bin`, `doc`, `include`, `src`, `obj` и содержать Makefile и README.md файлы.
15. Github Actions – функция, введенная в Github, позволяющая автоматизировать рабочий процесс. Она помогает при работе с различными процессами такими, как пушинг кода, создание релиза и т.д. Помогает при тестировании.
16. Workflow – это высокоуровневый набор правил для того, чтобы делать какие-то действия при определенных условиях. Репозитории могут содержать несколько workflows.
17. Event - действие, которое запускает рабочий процесс. Триггером может быть, например, `push` или `pull request`.
18. Job запускает какую-либо задачу в workflow. Изначально несколько задач выполняется параллельно, если между ними нет зависимости, в ином случае - последовательно.
19. Job разделяется на несколько шагов (step), которые являются либо действием, либо командой оболочки. Также изначально определена параллельность выполнения при отсутствии зависимостей.
20. Actions является часто используемым блоком кода. Каждое действие action может принимать на вход параметры и создавать любые значения, которые затем можно использовать в других действиях actions. Можно создавать собственные действия или использовать уже опубликованные.